# THE BIGMOTHER MANIFESTO: A ROADMAP TO SUPER-SAFE SUPER-BENEVOLENT SUPER-INTELLIGENT AGI (SUMMARY)

**Aaron Turner**
BigMother.AI CIC
Cambridge, UK
https://bigmother.ai
mailto:aaron.turner@bigmother.ai

Timestamp: 2023-03-23 13:10:45Z

## ABSTRACT

PLEASE NOTE: This document is currently undergoing major revision/refactoring (translation: it's a bit of a mess at the moment!) Please refer back occasionally (monthly...?) to see the latest updates.

***K*eywords** AGI · Artificial General Intelligence · cognitive architecture · neurosymbolic · super-intelligence · alignment

For Sir Clive Sinclair (30 July 1940 – 16 September 2021)

## Contents

# 1 It takes a village

Artificial General Intelligence (AGI) [1][2][3] is complicated. Firstly, the opportunities for miscommunication by an author, and misunderstanding by a reader, are endless. Secondly, it takes a village to build an AGI, and a particularly large and varied village to build a super-intelligent AGI [4][5]. Section 2 of the present paper is a summary of our work on AGI conducted from 1985 to 2023, but not previously reported[1]. An expanded version will follow in due course. The full paper will aim to be as accessible as possible to as wide and varied an audience as possible, i.e. the entire village[2].

# 2 BigMother

## 2.1 In a nutshell

The overall thrust of our argument is as follows:

1. Driven primarily by the unprecedented competitive advantage (both economic and strategic) to be gained from possessing more advanced AI than one's competitors, today's mostly-narrow AI will inexorably evolve, over the coming decades, first into increasingly general AI (AGI), and ultimately into super-intelligent agentic AGI.
2. Super-intelligence (i.e. super-intelligent agentic AGI) is a *trap-door* — once we've gone through it, it's not necesssarily the case that we will be able to go back. (Humanity trying to trick a super-intelligent agentic AGI that doesn't wish to be switched off into switching itself off is analogous to a cat trying to trick a human into not taking it to the vet!) Thus the nature of the *very first* super-intelligent[3] agentic AGI that mankind creates will likely determine the subsequent fate of all mankind for all eternity. We have exactly one chance to get it right.
3. Without intervention, mankind's current mostly-narrow-AI-to-super-intelligent-agentic-AGI trajectory [henceforth *AGI trajectory*], driven primarily by short-term self-interest, will most likely lead to an AGI endgame that is far from optimal for mankind as a whole, by which we mean anything from the extinction of the human race [6] to a world in which a handful of trillionaires enjoy exalted, near god-like, lives, while the vast majority exist in a continual nightmare of abject poverty and fear. It is vital that we overcome short-term self-interest!

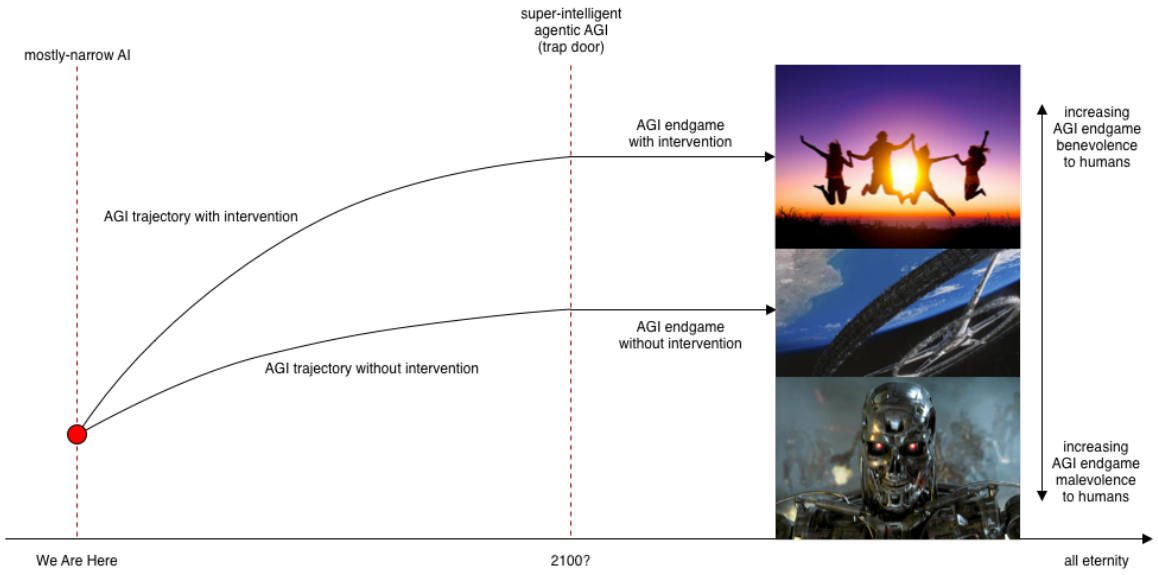


Figure 1: AGI trajectories and endgames

4. The goal of the BigMother project is to influence the AGI trajectory (and thus the AGI endgame, and thus the fate of all mankind for all eternity) in order to achieve an endgame that is safe and benevolent for all mankind.

[1] please note that any such summary must necessarily assume of the reader some prior understanding of the subject matter
[2] this is proving to be something of a Herculean task, so after a couple of years we decided to publish a summary first!
[3] or near-super-intelligent

5. In pursuance of 4, the BigMother project comprises two pillars:
   (a) The primarily non-technical **Governance** pillar. In order to thwart the worst instincts of, and to mitigate the worst harms caused by, those actors in the AGI field (including both corporations and nation states) who may be motivated primarily by short-term self-interest, it will be necessary for farsighted national, regional, and global policymakers to enact and then enforce appropriate AGI-related legislation, regulations, and treaties, including (i) measures designed to curtail unsafe AGI development, and (ii) measures designed to redistribute AGI-generated wealth. By far the greatest challenge will be to persuade the global population of AGI actors (including the policymakers themselves) to think beyond merely short-term self-interest. The BigMother project will seek to entreat policymakers to act in the best long-term interest of all mankind.
   (b) The primarily technical **Research and Development** pillar. The AGI policymakers' role will be greatly facilitated if they can be presented with a plausible — and overwhelmingly preferable — alternative to the default AGI trajectory. Our strategy therefore is to imagine *the ideal AGI endgame* and to then work backwards to achieve it, which is equivalent to imagining *the ideal super-intelligent agentic AGI* and working backwards to achieve that. Thus we seek to design, develop, and deploy a super-safe[4] super-benevolent[5] super-intelligent[6] alpha AGI[7] (called BigMother) that is publicly owned by all mankind, and whose operation benefits all mankind, without favouring any particular subset thereof (such as the citizens of any particular country or countries, or the shareholders of any particular company or companies).

The bulk of this paper is primarily concerned with 5b.

### 2.2 Pieces of the AGI jigsaw puzzle

#### 2.2.1 Problem-solving

Our objective is *utility* (i.e. that BigMother should exhibit intelligent behaviour that, from the human perspective, is both maximally safe and maximally benevolent). In pursuance of this objective, we equate *intelligence* with *problem-solving*.

This leads us to consider the space of all possible problems, which we define as follows: a problem is a triple $\langle R, Q, P \rangle$, where $R$ is a set of physical resource constraints (always including a finite time limit), $Q$ is an ordering over all possible things $T$, and $P$ is a property over possible things $T$. A triple $\langle R, Q, P \rangle$ should be interpreted by a *generic problem-solver* as follows: "using no more than resources $R$, strive to find some $Q$-minimal $x \in T$ such that $P(x)$"[8]. For example[9]:

- using no more than $S$ seconds of time or $J$ joules of energy, strive to find some $Q$-minimal thing $x$ such that $x \in \mathbb{N}$ and $x = 1 + 1$, where $Q$ is empty (any solution will do in the case that there's more than one solution)
- using no more than $S$ seconds of time or $J$ joules of energy, strive to find some $Q$-minimal thing $x$ such that $x \in \mathbb{Z}$ and $x^2 = 4$, where $Q$ favours positive solutions
- using no more than $S$ seconds of time or $J$ joules of energy, strive to find some $Q$-minimal thing $x$ such that $x \in \mathbb{N}$ and $\exists a, b, c, d \in \mathbb{N}^+ : \{a, b\} \neq \{c, d\} \wedge x = a^3 + b^3 \wedge x = c^3 + d^3$, where $Q$ favours smaller solutions.

Given an *arbitrary problem* $\langle R, Q, P \rangle$, a generic problem-solver:

- will strive to find the best (according to $Q$) solution (according to $P$) that it can, with no guarantees beyond that
- may deliver any of the following:
  - the best possible (i.e. maximum) solution, according to the specified (total) ordering
  - a best (i.e. maximal) solution, according to the specified (partial) ordering
  - a satisfactorily good solution, according to the specified ordering, albeit not quite the best
  - a disappointingly poor solution, according to the specified ordering, albeit not quite the worst
  - a worst possible (i.e. minimal) solution, according to the specified (partial) ordering
  - the worst possible (i.e. minimum) solution, according to the specified (total) ordering
  - no solution at all (e.g. if the problem has no solution, or if it runs out of resources before finding one).

When dealing with *arbitrary problems* chosen from the set of all possible problems, this is the best we can hope for. That said, some generic problem-solvers will deliver better (according to Q) solutions (according to P) than others.

[4] meaning (as a minimum) more safe than any human
[5] meaning (as a minimum) more benevolent than any human
[6] meaning (as a minimum) more intelligent than any human
[7] meaning (as a minimum) more intelligent than any other AGI with which it might coexist
[8] if $Q$ is empty then possible solutions are unordered, i.e. the problem being specified is not an optimisation problem
[9] these are all very simple examples; in principle, problems $\langle R, Q, P \rangle$ may be arbitrarily complex, i.e. as complex as the real world

### 2.2.2 The role of information in problem-solving

The key to intelligence is problem-solving, and the key to problem-solving is the effective use of information[10].

Thus there are (at least) two dimensions to a generic problem-solver (such as an AGI):

- the problem-solving mechanism *per se* (e.g. some combination of computer hardware and software)
- the information that *drives* that problem-solving mechanism towards solutions.

When designing a generic problem-solver (or a problem-solving mechanism within an AGI), we need to consider:

- how to *represent* the information needed for problem-solving (Section 2.2.3)
- how to *acquire* the information needed for problem-solving (Section 2.2.9)
- how to *use* the information so acquired to drive problem-solving (Section 2.2.10).

### 2.2.3 How to represent the information needed for problem-solving

We are building an AGI, with the emphasis on the G. Every component of an AGI needs to be as *general* as possible. Thus we need a representation mechanism with sufficient *richness of expression* to be able to capture "everything". We choose to use first-order set theory [7][8][9]; specifically, von Neumann–Bernays–Gödel set theory (NBG) [10].

First-order NBG set theory is *foundational*, meaning that *all of mathematics* [11] may be built on top of it. It also seems to be the case that NBG is sufficiently expressive to be able to capture the fine structure of *the entire physical universe*. Thus first-order NBG set theory satisfies our requirements for a representation mechanism for generic problem-solving.

The process through which we propose to construct first-order NBG set theory is as follows:

1. start with classical First-Order Logic with equality (FOL) [12][13][14] (Section 2.2.3.1):
   - FOL operates conceptually at the level of *belief* (i.e. declarative statements that are either true or false)[11]
   - *semantic consequence*[12] — arguably the most fundamental concept in logic — is foundational to FOL
   - *syntactic consequence*[13] — formal proof in FOL — is closely related to semantic consequence in FOL
   - in particular, FOL is both strongly sound[14] and strongly (semantically) complete[15] [13, p 35, p 39]
2. define a conservative extension[16] of FOL called UL-0, admitting the usual syntactic sugar (Section 2.2.3.2)
3. demonstrate that UL-0 is both strongly sound and strongly (semantically) complete[17] (Section 2.2.3.3)
4. strengthen UL-0 into UL-0-NBG (i.e. NBG set theory) by adding a set of axioms NBG (Section 2.2.3.4):
   - in UL-0-NBG, all sets are *classes*; some classes are sets; the remaining classes are called *proper classes*[18]
5. consider the logical consistency of the proposed UL-0-NBG axioms NBG[19] (Section 2.2.3.5)
6. define a basic mathematical toolkit (BMT) on top of UL-0-NBG (Section 2.2.3.6)
7. use UL-0-NBG (and the BMT) as the metalanguage in which to define the FOL UL-1 (Section 2.2.3.7)
8. strengthen UL-1 into UL-1-NBG by adding the exact same proper axioms as before (Section 2.2.3.8)
9. similarly construct an infinite stack of FOLs UL-N and set theories UL-N-NBG (Section 2.2.3.9)
10. define an extended mathematical toolkit (EMT) on top of UL-N-NBG (Section 2.2.3.10).

For most practitioners, the process of bootstrapping mathematics from the axioms up as described in Sections 2.2.3.1 to 2.2.3.10 is mind-numbingly tedious, boring, and dull. It is also an essential foundation for everything else that follows!

[10] every non-random problem-solving step reduces to some fragment of information justifying the evolution of the solution in a certain direction; in order to apply a sequence of steps, and reach a solution, a problem-solver must possess the necessary information

[11] we use the terms {belief, information, knowledge} interchangeably — philosophers should have a nap and a banana at this point!

[12] $C$ is a semantic consequence of $A_1...A_n$, written $A_1...A_n \vDash C$, iff $C$ is necessarily true whenever all of $A_1...A_n$ are true

[13] $C$ is a syntactic consequence of $A_1...A_n$, written $A_1...A_n \vdash C$, iff (the sequent) $A_1...A_n \vdash C$ is a *theorem* (see Section 2.2.3.1.3)

[14] if $A_1...A_n \vdash C$ then $A_1...A_n \vDash C$ (assuming that $A_1...A_n$ and $C$ are closed, i.e. contain no free variables)

[15] if $A_1...A_n \vDash C$ then $A_1...A_n \vdash C$ (assuming that $A_1...A_n$ and $C$ are closed)

[16] a *conservative extension* makes it easier to define, state, and prove theorems, but does not affect what is or is not a theorem

[17] in other words, as a quality control step, we must reliably ensure that we haven't "broken" FOL with our conservative extensions!

[18] a class is a collection of sets; a set is a class that is a member of some other class; a proper class is not a member of any class; proper classes are too big to be sets, e.g. the class of all sets $\mathbb{V}$, the class of ordinal numbers $\mathbb{On}$, or the class of surreal numbers $\mathbb{No}$

[19] if NBG $\vdash$ False is a theorem then so is NBG $\vdash$ "anything" and UL-0-NBG is unusable as a foundation for safe and benevolent AGI

#### 2.2.3.1 FOL

##### 2.2.3.1.1 Syntax

For our purposes, the syntax of FOL is defined by the following BNF [15] grammar:

$$
\begin{array}{rcl}
\langle \text{term} \rangle & ::= & \langle \text{variable} \rangle \\
& & \mid \langle \text{function symbol} \rangle \, \langle (\text{term*}) \rangle \\
\langle \text{wff} \rangle & ::= & \langle \text{relation symbol} \rangle \, \langle (\text{term*}) \rangle \\
& & \mid \wedge \langle (\text{wff*}) \rangle \\
& & \mid \vee \langle (\text{wff*}) \rangle \\
& & \mid \neg \langle \text{wff} \rangle \\
& & \mid ( \langle \text{wff} \rangle \Rightarrow \langle \text{wff} \rangle ) \\
& & \mid \Leftrightarrow \langle (\text{wff*}) \rangle \\
& & \mid ( \forall \langle \text{variable} \rangle : \langle \text{wff} \rangle ) \\
& & \mid ( \exists \langle \text{variable} \rangle : \langle \text{wff} \rangle ) \\
& & \mid = \langle (\text{term*}) \rangle
\end{array}
$$

$$
\begin{array}{rcl}
\langle \text{variable} \rangle & ::= & \textit{lowercase name} \\
\langle \text{function symbol} \rangle & ::= & \textit{lowercase name} \\
\langle \text{relation symbol} \rangle & ::= & \textit{lowercase name} \\
\langle \text{term+} \rangle & ::= & \langle \text{term} \rangle \mid \langle \text{term} \rangle \, , \, \langle \text{term+} \rangle \\
\langle \text{wff+} \rangle & ::= & \langle \text{wff} \rangle \mid \langle \text{wff} \rangle \, , \, \langle \text{wff+} \rangle \\
\langle \text{term*} \rangle & ::= & \epsilon \mid \langle \text{term+} \rangle \\
\langle \text{wff*} \rangle & ::= & \epsilon \mid \langle \text{wff+} \rangle \\
\langle (\text{term*}) \rangle & ::= & ( \langle \text{term*} \rangle ) \\
\langle (\text{wff*}) \rangle & ::= & ( \langle \text{wff*} \rangle )
\end{array}
$$

Note that, in our formulation:

- variables, function symbols, and relation (i.e. predicate) symbols are all lowercase names
- a ⟨term+⟩ is a comma-separated list of one or more ⟨term⟩s (similarly re ⟨wff+⟩)
- a ⟨term*⟩ is a comma-separated list of zero or more ⟨term⟩s (similarly re ⟨wff*⟩)
- a ⟨(term*)⟩ is a comma-separated list of zero or more ⟨term⟩s, in parentheses (similarly re ⟨(wff*)⟩)
- a ⟨term⟩ comprising a ⟨function symbol⟩ followed by empty parentheses denotes a constant
- a ⟨wff⟩ comprising a ⟨relation symbol⟩ followed by empty parentheses denotes a proposition
- ∧, ∨, ⇔, and = are n-ary operators, not binary
- ∧(), ⇔(), and =() denote True, whereas ∨() denotes False.

##### 2.2.3.1.2 Semantics

It is a straightforward matter to assign a standard semantics (in the style of Tarski [16][17]) to our formulation of FOL:

- ⟨variable⟩s denote individuals (i.e. members of the domain of discourse)
- ⟨function symbol⟩s denote (possibly-nullary) functions over individuals
- ⟨relation symbol⟩s denote (possibly-nullary) relations over individuals
- ∧⟨(wff*)⟩ denotes True iff every $W \in \langle \text{wff*} \rangle$ denotes True
- ∨⟨(wff*)⟩ denotes True iff some $W \in \langle \text{wff*} \rangle$ denotes True
- ¬⟨wff⟩ denotes True iff ⟨wff⟩ denotes False
- $(\langle \text{wff} \rangle_1 \Rightarrow \langle \text{wff} \rangle_2)$ denotes True iff $\langle \text{wff} \rangle_1$ denotes False or $\langle \text{wff} \rangle_2$ denotes True
- ⇔ ⟨(wff*)⟩ denotes True iff every $W \in \langle \text{wff*} \rangle$ denotes True or every $W \in \langle \text{wff*} \rangle$ denotes False
- (∀⟨variable⟩ : ⟨wff⟩) denotes True iff ⟨wff⟩ denotes True for every safely-substituted possible value of ⟨variable⟩
- (∃⟨variable⟩ : ⟨wff⟩) denotes True iff ⟨wff⟩ denotes True for some safely-substituted possible value of ⟨variable⟩
- =⟨(term*)⟩ denotes True iff $T_1 = T_2$ (in the metalanguage) for every $T_1, T_2 \in \langle \text{term*} \rangle$.

#### 2.2.3.1.3 Proof

A *theorem* of FOL is any sequent $A_1 ... A_n \vdash C$ derivable via the rules of natural deduction [18][19], as follows:

Table 1: Natural deduction inference rules for FOL

| rule-name | $\frac{premises}{conclusion}$ | side-condition | rule-name | $\frac{premises}{conclusion}$ | side-condition |
|---|---|---|---|---|---|
| Hyp | $\frac{}{As \vdash C}$ | $C \in As$ | Weaken | $\frac{As \vdash C}{Bs \vdash C}$ | $As \subseteq Bs$ |
| $\wedge$-intro | $\frac{As \vdash C \text{ for all } C \in Cs}{As \vdash \wedge(Cs)}$ | | $\wedge$-elim | $\frac{As \vdash \wedge(Cs)}{As \vdash C}$ | $C \in Cs$ |
| $\vee$-intro | $\frac{As \vdash \vee(Cs)}{As \vdash \vee(Ds)}$ | $Cs \subseteq Ds$ | $\vee$-elim | $\frac{As \vdash \vee(Cs) \quad As \vdash (C \Rightarrow D) \text{ for all } C \in Cs}{As \vdash D}$ | |
| $\neg$-intro | $\frac{As \vdash (C \Rightarrow D) \quad As \vdash (C \Rightarrow \neg D)}{As \vdash \neg C}$ | | $\neg$-elim | $\frac{As \vdash \neg\neg C}{As \vdash C}$ | |
| $\Rightarrow$-intro | $\frac{As^\frown C \vdash D}{As \vdash (C \Rightarrow D)}$ | | $\Rightarrow$-elim | $\frac{As \vdash C \quad As \vdash (C \Rightarrow D)}{As \vdash D}$ | |
| $\Leftrightarrow$-intro | $\frac{As \vdash (C \Rightarrow D) \text{ for all } C, D \in Cs}{As \vdash \Leftrightarrow(Cs)}$ | | $\Leftrightarrow$-elim | $\frac{As \vdash \Leftrightarrow(Cs)}{As \vdash (C \Rightarrow D)}$ | $C, D \in Cs$ |
| $\forall$-intro | $\frac{As \vdash C[T \text{ for all } V]}{As \vdash (\forall\, V : C)}$ | $T \notin As, T \notin C$ | $\forall$-elim | $\frac{As \vdash (\forall\, V : C)}{As \vdash C[T \text{ for all } V]}$ | |
| $\exists$-intro | $\frac{As \vdash C[T \text{ for all } V]}{As \vdash (\exists\, V : C)}$ | | $\exists$-elim | $\frac{As \vdash (\exists\, V : C) \quad As \vdash (\forall\, V : (C \Rightarrow D))}{As \vdash D}$ | $V$ not free in $D$ |
| $=$-intro | $\frac{}{As \vdash =(U)}$ | | $=$-elim | $\frac{As \vdash =(Us) \quad As \vdash C}{As \vdash C[\text{some } Us \text{ for some } Us]}$ | $Us$ is non-empty |

In Table 1: $As, Bs, Cs, Ds$ are ⟨wff*⟩s; $C, D$ are ⟨wff⟩s; $As^\frown C$ denotes concatenation; $V$ is a ⟨variable⟩; a variable is free if it is not bound by any variable-binding operator (such as $\forall$ or $\exists$); $T, U$ are ⟨term⟩s; $T$ is closed; $Us$ is a ⟨term*⟩; $C[T \text{ for all } V]$ denotes the ⟨wff⟩ obtained by substituting $T$ for all free occurrences of $V$ in $C$[20]; $C[\text{some } Us \text{ for some } Us]$ denotes any ⟨wff⟩ obtained by safely substituting some $U_1 \in Us$ for some occurrences of $U_2 \in Us$ in $C$[21]. (Phew!)

[20] note that, in this instance, because $T$ is closed (contains no free variables), unintended variable capture cannot occur

[21] renaming bound variables as necessary in order to avoid unintended variable capture (see e.g. [13][p 22])

#### 2.2.3.2 UL-0

FOL may be foundational in principle, but nevertheless we will need to add a few conservative extensions — *indefinite descriptions*, *definite descriptions*, and *named definitions* — before being able to use it for any serious mathematics.

We shall refer to the resulting language as UL-0[22].

##### 2.2.3.2.1 Indefinite descriptions

Following Russell's *Theory of Descriptions* [20][21][22], we extend FOL with *indefinite descriptions* as follows:

The UL-0 ⟨wff⟩

$G(?x : F(x))$ [read: $G$ is True for *some* $x$ for which $F(x)$ is True]

(where $F$ and $G$ are unary ⟨relation symbol⟩s) is equivalent to the FOL ⟨wff⟩

$(\exists x : \wedge(F(x),\ G(x)))$

which reads:

- there exists some x for which $F(x)$ is True
- and $G(x)$ is also True for that x.

Thus we are justified in adding two new inference rules — ?-intro and ?-elim — to the UL-0 proof system.

Note that indefinite descriptions may be readily extended to n-ary ⟨relation symbol⟩s, i.e. $G(..., ?x : F(..., x, ...), ...)$.

##### 2.2.3.2.2 Definite descriptions

Again following Russell's *Theory of Descriptions*, we extend FOL with *definite descriptions* as follows:

The UL-0 ⟨wff⟩

$G(!x : F(x))$ [read: $G$ is True for *the* $x$ for which $F(x)$ is True]

(where $F$ and $G$ are unary ⟨relation symbol⟩s) is equivalent to the FOL ⟨wff⟩

$(\exists x : \wedge(F(x),\ (\forall y : (F(y) \Rightarrow = (x, y))),\ G(x)))$

which reads:

- there exists some x for which $F(x)$ is True
- and that x is unique
- and $G(x)$ is also True for that x.

Thus we are justified in adding two further new inference rules — !-intro and !-elim — to the UL-0 proof system.

Note that definite descriptions may also be readily extended to n-ary ⟨relation symbol⟩s, i.e. $G(..., !x : F(..., x, ...), ...)$.

##### 2.2.3.2.3 Named definitions

Simply stated, a UL-0 *script* comprises a sequence of *definitions*, each of which comprises:

- one or more keywords specifying the *type* of the definition
- the definition's *signature*
- a double-equals (==) denoting "is defined by"
- the *body* of the definition
- a semicolon

in that order.

UL-0 definitions are described in Sections 2.2.3.2.4 to 2.2.3.2.15.

[22] where "UL" stands (rather immodestly!) for "Universal Logic"

#### 2.2.3.2.4 Wff definitions

A `$wff` definition defines a new ⟨wff⟩ form (i.e. a new name of syntactic type ⟨wff⟩). For example:

```
$wff {[@T1] bar [@T2]}
== bar(@T1, @T2);
```

defines:

```
{[] bar []}
```

as a new ⟨wff⟩ form[23], as follows:

- `$wff` indicates that this is a `$wff` definition
- `{[@T1] bar [@T2]}` is the definition *signature*
  - the enclosing `{` and `}` indicate that this signature is of syntactic type ⟨wff⟩
  - `[@T1]` and `[@T2]` are parameter lists
  - the enclosing `[` and `]` indicate that each of these parameter lists takes a single ⟨term⟩ parameter
  - `@T1` and `@T2` are *formal parameters*, each placeholders (in this case) for a single ⟨term⟩
  - formal parameter names are always prefixed by `@`, and are always in upper case
  - the *name* being defined here is `{[] bar []}`, i.e. the signature without the formal parameters
  - the name introduced by each new definition must be unique (i.e. not already defined)
- `bar(@T1, @T2)` is the definition body
- the formal parameters defined in the signature (here `@T1` and `@T2`) may appear in the definition body; a ⟨term⟩ formal parameter may appear anywhere that a ⟨term⟩ may appear (similarly re ⟨wff⟩ etc formal parameters).

#### 2.2.3.2.5 Definition application

Once a new name of syntactic type S has been defined, its signature — with formal parameters replaced by actual parameters of the appropriate syntactic types — may appear anywhere that an expression of syntactic type S may appear.

For example, given the above definition of `{[] bar []}`:

```
{[a()] bar [f(g(x), y)]}
```

may appear anywhere that a ⟨wff⟩ may appear; `a()` and `f(g(x), y)` are *actual parameters* of syntactic type ⟨term⟩.

#### 2.2.3.2.6 Introduction and elimination of defined names

A definition application (such as `{[a()] bar [f(g(x), y)]}`) is semantically equivalent to the corresponding definition body with the specified actual parameters *safely substituted* for the corresponding formal parameters[24].

For example:

```
{[a()] bar [f(g(x), y)]}
```

is semantically equivalent to:

```
bar(a(), f(g(x), y)).
```

As a result of this equivalence, each definition D effectively introduces a new natural deduction inference rule (such as `{[] bar []}`-intro) allowing defined names to be *introduced* during the proof process, as well as a new natural deduction inference rule (such as `{[] bar []}`-elim) allowing defined names to be *eliminated* during the proof process.

[23] subsequent to this definition, syntactic expressions having this syntactic form may appear anywhere that a ⟨wff⟩ may appear

[24] the usual care must be taken to rename bound variables as required in order that inadvertent variable capture cannot occur

#### 2.2.3.2.7 Parameter list types

The above definition of `{[] bar []}` has two parameter lists, each enclosed in `[` and `]` indicating that each of these parameter lists expects a single ⟨term⟩ parameter, and that `@T1` and `@T2` are each placeholders for a single ⟨term⟩.

Other parameter list types are possible, as follows:

Table 2: UL-0 parameter list types

| parameter list brackets | formal parameter type | actual parameter type |
|---|---|---|
| `[` and `]` | ⟨term⟩ | ⟨term⟩ |
| `{` and `}` | ⟨wff⟩ | ⟨wff⟩ |
| `[*` and `*]` | ⟨term*⟩ | ⟨term*⟩ |
| `{*` and `*}` | ⟨wff*⟩ | ⟨wff*⟩ |
| < and > | ⟨variable⟩ | ⟨variable⟩ |

For example:

```
$wff {foo {*@WS*}}
== foo(@WS);
```

defines the new name `{foo {**}}` such that `@WS` is a formal parameter of type ⟨wff*⟩, and the definition instance:

```
{foo {* {[x] bar [y]}, {[a()] bar [f(g(x), y)]} *}}
```

expands into:

```
foo({[x] bar [y]}, {[a()] bar [f(g(x), y)]}).
```

#### 2.2.3.2.8 Separators

As a little syntactic sugar, UL-0 allows names taking a single ⟨term*⟩ or ⟨wff*⟩ to be defined as *separators*; for example:

```
$wff {{*@WS*} $separator foo}
== foo(@WS);
```

defines the new name `{{**} $separator foo}` such that `@WS` is a formal parameter of type ⟨wff*⟩.

Given the definition instance:

```
{     {[x] bar [y]}
  foo {[a()] bar [f(g(x), y)]}
}
```

the ⟨wff⟩ actual parameters:

```
{[x] bar [y]}
```

and:

```
{[a()] bar [f(g(x), y)]}
```

are assembled into the ⟨wff*⟩ actual parameter:

```
{[x] bar [y]}, {[a()] bar [f(g(x), y)]}
```

which is then substituted for the ⟨wff*⟩ formal parameter `@WS` in the definition body, giving:

```
foo({[x] bar [y]}, {[a()] bar [f(g(x), y)]}).
```

*** OK TO HERE 03:29 20.3.23

#### 2.2.3.2.9 Term definitions

`$the`

`$some`

*** TODO

#### 2.2.3.2.10 Guards

*** TODO

#### 2.2.3.2.11 Fresh variables

*** TODO

#### 2.2.3.2.12 Variable-binding operators

*** TODO

#### 2.2.3.2.13 Conjectures, proof obligations, and proofs

*** TODO

*** UL-0 turnstiles must be decorated with "defs" to indicate that the consequence depends on the current definitions

#### 2.2.3.2.14 Axioms

*** TODO

#### 2.2.3.2.15 Example definitions: Literate FOL

```
$wff {{*@WS*} $separator and}
== ∧(@WS);

$wff {{*@WS*} $separator or}
== ∨(@WS);

$wff {not {@W}}
== ¬@W;

$wff {if {@W1} then {@W2}}
== (@W1 ⇒ @W2);

$wff {{*@WS*} $separator iff}
== ⇔(@WS);

$wff $vbo {for-all <$binding @V> :  {$scope @W}}
== (∀ @V : @W);

$wff $vbo {for-some <$binding @V> :  {$scope @W}}
== (∃ @V : @W);

$wff {[*@TS*] $separator =}
== =(@TS);

$wff {[@T1] != [@T2]}
== {not {=(@T1, @T2)}};
```

#### 2.2.3.3 Quality control: UL-0 soundness and completeness

*** TODO

#### 2.2.3.4 UL-0-NBG

Even with indefinite descriptions, definite descriptions, and named definitions, UL-0 still isn't sufficiently expressive for our purposes. In particular, any function or relation symbols appearing in a semantic consequence $A_1...A_n \vDash_{\texttt{defs}} C$ lack any *specific* meaning, and thus $A_1...A_n \vDash_{\texttt{defs}} C$ is only valid if $C$ is necessarily true whenever all of $A_1...A_n$ are true for every possible interpretation (i.e. specific meaning) of the function and relation symbols appearing within it.

One solution to this problem is to add additional assumptions — known as *proper axioms* — to the left of the turnstile, such that all semantic consequences in the resulting *first-order theory* have the form $\texttt{AXIOMS}^\frown A_1...A_n \vDash_{\texttt{defs}} C$.

The net effect of the `AXIOMS` is to eliminate from consideration any semantic interpretation in which the functions assigned to certain ⟨function symbol⟩s and the relations assigned to certain ⟨relation symbol⟩s fail to make the `AXIOMS` true. In other words, in any such theory, we only care about those semantic interpretations that make the `AXIOMS` true.

In order to strengthen UL-0 into the NBG set theory UL-0-NBG, therefore, we need to define a set of proper axioms `NBG` having the collective effect of constraining the meaning of the binary relation $in(x, y)$ to "class membership", i.e. $x \in y$.

Luckily, the axioms of NBG (some of which are effectively optional) have already been formulated by various authors; we just need to select a suitable subset and define them in UL-0. For our purposes, we will use the following axioms:

- the Extensionality Axiom[25]
- the Pairing Axiom[26]
- the Class Existence Axioms[27]
- the Power Set Axiom[28]
- the Sum Set Axiom[29]
- the Axiom of Infinity[30]
- the Limitation of Size Axiom[31]
- the Axiom of Regularity[32].

[25] ensures that two classes are equal iff they contain exactly the same members; also implies that a unique empty class exists

[26] ensures the existence of sets containing either one or two members

[27] ensures the existence of various classes formed in various ways (e.g. via intersection, complement, Cartesian product, etc); most importantly, the Class Existence Axioms allow us to define class comprehension, i.e. "the class of all sets $x$ such that $P(x)$"

[28] ensures that the power class of a set is a set

[29] ensures that the sum class of a set is a set

[30] ensures that an infinite set exists

[31] implies the Axiom of Replacement, the Axiom of Separation, and the Global Choice Axiom, and also ensures (i) that the proper classes are exactly those that are equinumerous with the class of all sets $\mathbb{V}$, (ii) that $\mathbb{V}$ is equinumerous with the class of all ordinal numbers, (iii) that $\mathbb{V}$ can be well-ordered, and (iv) that every cardinal number may be identified with its corresponding initial ordinal

[32] implies that (i) no set is an element of itself, (ii) for any two sets, at most one is an element of the other, (iii) there are no infinitely descending membership sequences, (iv) every set has an ordinal rank $r \in \mathbb{O}\mathbb{m}$, and (v) $\mathbb{V}$ = the von Neumann hierarchy of sets $\mathbb{H}$

#### 2.2.3.4.1 Example definitions: Literate set theory

```
$wff {[@T1] is-in [@T2]}
== in(@T1, @T2);

$wff {[@T] is-a-set}
$assuming $fresh @V
== {for-some <@V>
   : {[@T] is-in [@V]}
   };

$wff $vbo {for-all-sets <$binding @V> : {$scope @W}}
== {for-all <@V>
   : {if   {[@V] is-a-set}
      then {@W}
     }
   };

$wff $vbo {for-some-set <$binding @V> : {$scope @W}}
== {for-some <@V>
   : {    {[@V] is-a-set}
      and {@W}
     }
   };

$axiom {extensionality}
== {for-all <x>
   : {for-all <y>
     : {    {[x] = [y]}
        iff {for-all <z>
            : {    {[z] is-in [x]}
               iff {[z] is-in [y]}
              }
            }
       }
     }
   };

$wff {the-members-of [@T] are-exactly [@T1] and [@T2]}
$assuming $fresh @V
== {for-all-sets <@V>
   : {    {[@V] is-in [@T]
      iff {   {[@V] = [@T1]}
           or {[@V] = [@T2]}
          }
     }
   };

$axiom {pairing}
== {for-all-sets <x>
   : {for-all-sets <y>
     : {for-some-set <z>
       : {the-members-of [z] are-exactly [x] and [y]}
       }
     }
   };

$the [unordered-pair [@T1], [@T2]]
$where {{[@T1] is-a-set} and {[@T2] is-a-set}}
$assuming $fresh @V
== !@V : {the-members-of [@V] are-exactly [@T1] and [@T2]};
```

```
$the [ordered-pair [@T1], [@T2]]
$where {{[@T1] is-a-set} and {[@T2] is-a-set}}
$assuming $fresh @V
== [unordered-pair [@T1], [unordered-pair [@T1], [@T2]]];
```

#### 2.2.3.5 Quality control: Consistency of the UL-0-NBG axioms

*** TODO

#### 2.2.3.6 Basic mathematical toolkit

*** TODO

in NBG, both the complement of the empty set and the generalised intersection of the empty set may be defined

#### 2.2.3.7 UL-1

*** TODO

#### 2.2.3.8 UL-1-NBG

*** TODO

#### 2.2.3.9 UL-N and UL-NBG-N

*** TODO

#### 2.2.3.10 Extended mathematical toolkit

*** TODO

### 2.2.4 The high-level cognitive architecture of an AGI

At this point, it will be instructive to consider the high-level *cognitive architecture* of an AGI M:

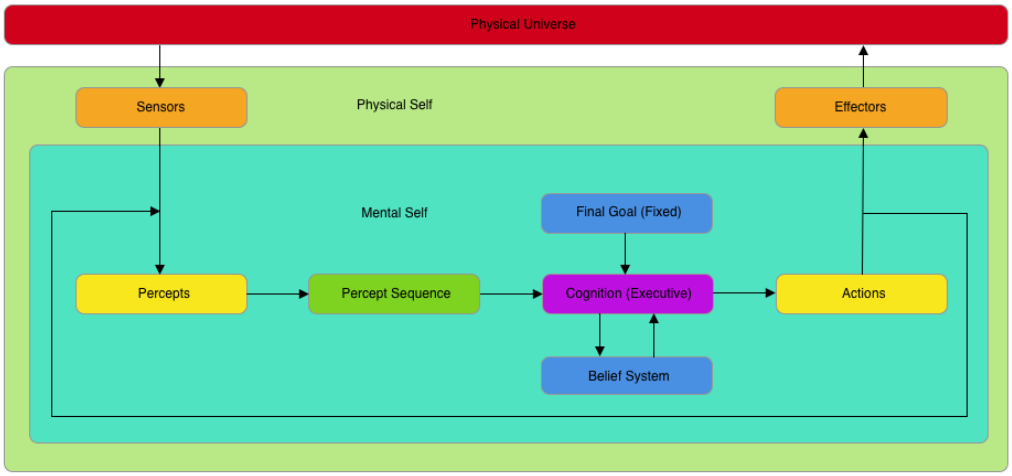


Figure 2: High-level cognitive architecture of an AGI

Looking at Figure 2:

- M *observes* the physical universe via various *sensors*
- each sensor generates a stream of *percepts*, each of which contains a discrete packet of information[33]
- percepts are collected together into a *percept sequence* containing M's entire history of perceptual experiences[34]
- M's *cognition*:
  - is effectively the "Executive" (computer program) controlling the entire system
  - maintains a *belief system* (set of *beliefs*[35]), corresponding to the set of things that M currently believes (including everything that M believes about the universe — effectively, M's *internal model* of the universe)
  - generates a stream of *actions*, most (but not necessarily all) of which drive an *effector*
  - strives to interact with the universe (via M's effectors) in such a way as to realise its (fixed) *final goal*
- note that actions are *looped back* as percepts; thus M's cognition is able to immediately observe its own actions
- in many cases, M's interactions with the physical universe (via M's effectors) will causally affect the universe in some way that M is subsequently able to observe (via M's sensors), albeit after some positive time delay
- thus any observable effect of an action will appear *later* in the percept sequence than the action that caused it
- some of M's sensors are *internal* in nature[36]; thus:
  - M's cognition is able to sense M's *internal physical self* (e.g. detecting temperature)
- some of M's effectors are also internal in nature; thus:
  - M's cognition is able to affect M's internal physical self (e.g. adjusting fan speed)
- M's cognition is able to inject *imagined experiences* into its own percept sequence[37]; thus:
  - M's cognition is able to both affect and sense M's *internal mental self*.

[33] any percept, derived from any sensor, may be logically represented as a finite graph [23] (which may be represented in UL-N-NBG)

[34] a monotonically-growing percept sequence may be logically represented as a sequence of graphs, indexed by $pseudotime \in \mathbb{N}$

[35] for our purposes, each *belief* is a sequent of the form $\texttt{NBG}^\frown A_1 ... A_n \vDash \texttt{defs}\ C$ for which the sequent $\texttt{NBG}^\frown A_1 ... A_n \vdash \texttt{defs}\ C$ is a theorem (via natural deduction); a belief's syntactic structure (its abstract syntax tree) may be logically represented as a finite graph

[36] M's physical self is necessarily a part of the physical universe

[37] for our purposes, we will assume that (a) each sensor has a unique integer ID $< 0$, (b) each effector has a unique integer ID $> 0$, (c) each action is tagged with an integer ID $\geq 0$, and (d) percepts are tagged with the ID of the sensor ($< 0$) or action ($\geq 0$) from which they originate; if we now allow M's cognition to generate actions with ID $= 0$, such actions don't relate to any effector, so they don't affect the physical universe in any way, but a *copy* of any such action is nevertheless looped back as an *imagined percept*

#### 2.2.4.1 Cognitive processes

If we expand the *cognition* component of Figure 2, we will see that it comprises a number of *cognitive processes*:

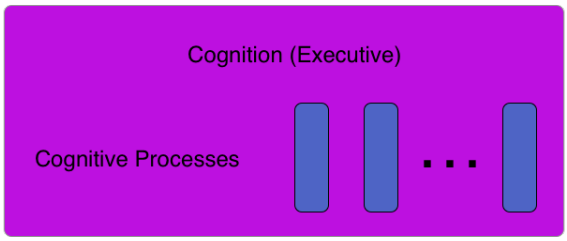


Figure 3: Cognitive processes

For our purposes, each cognitive process is a *non-terminating computer program* executing in parallel with the others.
The number of cognitive processes, and their functionalities, is determined by the AGI designer.

#### 2.2.4.2 Cognitive primitives

If we expand the *cognition* component of Figure 2 still further, we will see a number of *cognitive primitives*[38]:

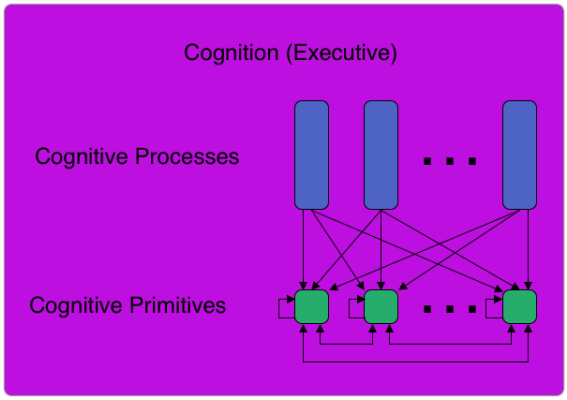


Figure 4: Cognitive primitives

For our purposes, each cognitive primitive is a *program subroutine* implementing some specific functionality.
Each cognitive process is implemented via calls to one or more cognitive primitives, which may also invoke each other.
The number of cognitive primitives, and their functionalities, is determined by the AGI designer.
For our purposes, we will focus (initially) on the following cognitive primitives:

- verification (Section 2.2.5)
- deduction (Section 2.2.6)
- abduction (Section 2.2.7)
- induction (Section 2.2.8)

the first three of which follow immediately from the definition of semantic consequence (Section 2.2.3).

[38] effectively an API for AGI

#### 2.2.5 Verification

We imagine that some cognitive process:

- has formulated closed ⟨wff⟩s $A_1$ ... $A_n$
- has formulated closed ⟨wff⟩ $C$
- needs to determine whether or not $A_1$ ... $A_n \vDash C$.

Accordingly, we define the *verification* cognitive primitive as follows:

**Inputs**:

(1) $R$: a set of physical resource constraints (always including a finite time limit)
(2) closed ⟨wff⟩s $A_1$ ... $A_n$
(3) closed ⟨wff⟩ $C$
(4) an optional blackboard (which, if present, may be used as a starting point)

**Outputs**:

(1) one of:
(a) an indication that some of the resources specified by $R$ have been depleted
(b) a pair $\langle x, y \rangle$, where:
- x is a proof of $A_1$ ... $A_n \vDash C$
- y is a proof that x is a proof of $A_1$ ... $A_n \vDash C$

(c) a proof of the non-existence of any proof of $A_1$ ... $A_n \vDash C$

(2) a blackboard (containing its deliberations so far)

**Behaviour**:

verification will strive to deliver (via output (1)(b)) the shortest proof $x$ that it can

**Implementation**:

*** TODO

##### 2.2.5.1 Iterated verification

*** TODO

#### 2.2.6 Deduction

We imagine that some cognitive process:

- has formulated closed ⟨wff⟩s $A_1$ ... $A_n$
- needs to determine what closed ⟨wff⟩s $C$ are logically entailed by (i.e. *necessarily follow from*) $A_1$ ... $A_n$.

In practice:

- there may be zero, one, or many (even infinitely many) closed ⟨wff⟩s $C$ for which $A_1$ ... $A_n \vDash C$
- it will often be appropriate, therefore, to restrict the results to a specific subset-of-interest (called the *focus*).

Accordingly, we define the *deduction* cognitive primitive as follows:

**Inputs**:

(1) $R$: a set of physical resource constraints (always including a finite time limit)
(2) closed ⟨wff⟩s $A_1$ ... $A_n$
(3) $F$: a property over closed ⟨wff⟩s (effectively, the current focus)
(4) an optional blackboard (which, if present, may be used as a starting point)

**Outputs**:

(1) one of:
  (a) an indication that some of the resources specified by $R$ have been depleted
  (b) a pair $\langle x, y \rangle$, where:
    - x is a non-empty set of closed ⟨wff⟩s $C$ for which $(A_1 \text{ ... } A_n \vDash C) \wedge F(C)$
    - $y$ is a proof that $(A_1 \text{ ... } A_n \vDash C) \wedge F(C)$ holds for each $C \in x$
  (c) a proof of the non-existence of any closed ⟨wff⟩s $C$ for which $(A_1 \text{ ... } A_n \vDash C) \wedge F(C)$
(2) a blackboard (containing its deliberations so far)

**Behaviour**:

deduction will strive to deliver (via output (1)(b)) the largest set $x$ that it can

**Implementation**:

*** TODO

##### 2.2.6.1 Iterated deduction

*** TODO

#### 2.2.7 Abduction

We imagine that some cognitive process:

- has formulated closed ⟨wff⟩s $A_1$ ... $A_n$
- has formulated closed ⟨wff⟩ $C$
- has (possibly) already determined (via verification) that $A_1$ ... $A_n \not\models C$
- needs to determine what closed ⟨wff⟩s $H$, if appended to $A_1$ ... $A_n$, would logically entail (i.e. *explain*) $C$.

In practice:

- there may be zero, one, or many (even infinitely many) closed ⟨wff⟩s $H$ for which $A_1$ ... $A_n$, $H \models C$
- it will often be appropriate, therefore, to restrict the results to a specific subset-of-interest (called the *focus*).

Accordingly, we define the *abduction* cognitive primitive as follows:

**Inputs**:

(1) $R$: a set of physical resource constraints (always including a finite time limit)
(2) closed ⟨wff⟩s $A_1$ ... $A_n$
(3) closed ⟨wff⟩ $C$
(4) $F$: a property over closed ⟨wff⟩s (effectively, the current focus)
(5) an optional blackboard (which, if present, may be used as a starting point)

**Outputs**:

(1) one of:
  (a) an indication that some of the resources specified by $R$ have been depleted
  (b) a pair $\langle x, y \rangle$, where:
    - x is a non-empty set of closed ⟨wff⟩s $H$ for which ($A_1$ ... $A_n$, $H \models C$) $\wedge F(H)$
    - $y$ is a proof that ($A_1$ ... $A_n$, $H \models C$) $\wedge F(H)$ holds for each $H \in x$
  (c) a proof of the non-existence of any $H$ for which ($A_1$ ... $A_n$, $H \models C$) $\wedge F(H)$

(2) a blackboard (containing its deliberations so far)

**Behaviour**:

abduction will strive to deliver (via output (1)(b)) the largest set $x$ that it can

**Implementation**:

*** TODO

##### 2.2.7.1 Iterated abduction

*** TODO

#### 2.2.8 Induction

*** TODO

*** (Grenander-Mumford pattern theory [24][25][26][27][28][29])

##### 2.2.8.1 Iterated induction

*** TODO

#### 2.2.9 How to acquire the information needed for problem-solving

There are three ways in which an AGI may acquire information:

- **innate information** — "built-in" or "given" information, hand-coded by its designers[39]
- **experiential information** — information acquired by direct experience[40]
- **latent information** — information derived from other information via calculation[41].

*** TODO

##### 2.2.9.1 belief maintenance

*** TODO

*** metaphysical stuff

*** abstract mathematical universe, concrete physical universe

*** problem-solving (the essence of intelligence) = cognition (the algorithmic part) + knowledge (the information part)

*** it's incredibly important therefore to have a DEEP understanding rather than a SHALLOW understanding

*** DEEP understanding = more and better information to use for problem-solving purposes -> better solutions

*** DEEP understanding of the physical universe -> better solutions to problems pertaining to the physical universe

*** the internal model of the PU constructed by induction must accurately discover the DEEP structure of the PU

*** Unified Belief Theory (UBT) — iterated induction-followed-by-abduction

#### 2.2.10 How to use the information so acquired to drive problem-solving

##### 2.2.10.1 Invuction

Following Section 2.2.1, we define a new cognitive primitive — *invuction*[42] — as follows:

**Inputs**:

(1) $R$: a set of physical resource constraints (always including a finite time limit)
(2) $Q$: an ordering over sets $s \in \mathbb{V}$
(3) $P$: a property over sets $s \in \mathbb{V}$
(4) an optional blackboard (which, if present, may be used as a starting point)

**Outputs**:

(1) one of:
  (a) an indication that some of the resources specified by $R$ have been depleted
  (b) a pair $\langle x, y \rangle$, where:
    - $P(x)$
    - $y$ is a proof of $P(x)$
  (c) a proof of the non-existence of any set $x$ for which $P(x)$

(2) a blackboard (containing its deliberations so far)

**Behaviour**:

invuction will strive to deliver (via output (1)(b)) the best (according to $Q$) $x$ that it can

**Implementation**:

*** TODO

[39] e.g. the BMT (Section 2.2.3.6), the EMT (Section 2.2.3.10), or the Final Goal (Section 2.2.4, Figure 2)
[40] i.e. percepts in the percept sequence (Section 2.2.4, Figure 2) tagged with an ID $< 0$, $= 0$, or $> 0$
[41] e.g. via verification (Section 2.2.5), deduction (Section 2.2.6), abduction (Section 2.2.7), and/or induction (Section 2.2.8)
[42] from the Latin *invenire*: to find, discover, invent, devise

#### 2.2.10.2 Iterated invuction

*** TODO

#### 2.2.10.3 Invuction is all you need

*** TODO

### 2.2.11 Quality control: Well-founded AGI

*** TODO

*** HW and SW synthesis of major AGI components (yielding both source code and formal proofs of correctness)

## 2.3 Putting it all together

*** TODO

*** intelligence = cognition (the algorithmic part) + knowledge (the information part)

*** TF super-intelligent = super-cognitive + super-knowledgeable

*** super-cognitive means being a better problem-solver than any human

*** super-knowledgeable means being more knowledgeable than any human

*** super-safe means safer than any human

*** super-benevolent means more benevolent than any human

*** super-safe/benevolent = maximally-aligned + super-cognitive + super-knowledgeable

*** maximally safe/benevolent = maximally-aligned + maximally cognitive + maximally knowledgeable

*** safety sequencing

*** start with top-level goal as defined in previous section; proceed via top-down refinement

*** alpha AGI design principles (derived from earlier considerations of problem-solving, human cognitive flaws etc)

*** maximal alignment — assuming super-cognitive and super-knowledgeable (all human knowledge, inc AI safety):

"Perform Directives D(1)-(2) (simultaneously, and to the best of your ability), subject to Qualification Q(a), as follows:

- D(1) for each individual human being (living or future), strive to estimate (as accurately as possible) their actual well-informed freely-chosen preferences
- D(2) for each individual human being (living or future), strive to maximise the extent to which their actual well-informed freely-chosen preferences (as you have estimated them) are (and likely will be) realised;
- Q(a) use your estimation of the actual well-informed freely-chosen preferences of the living human being population to resolve (to the best of your ability) any trade-offs that may arise in respect of D(1)-(2)."[43]

*** super-cognitive — assumed by above final goal; inductive-deductive-abductive (IDA) problem-solving architecture

****** IDA: maintains percept sequence (ordered history of all percepts sampled from sensors)

****** IDA: maintains belief system (all beliefs are theorems of first-order NBG set theory)

****** IDA: initial beliefs (toolbox of theorems, inc e.g. probability and statistics) hand-coded by designers

****** IDA: additional beliefs (all of which are theorems) continually synthesised by induction on percept sequence

****** IDA: problem-solving proceeds by applying deduction and abduction to beliefs (theorems) in the belief system

*** super-knowledgeable — assumed by above final goal; assumes working induction; take the AGI to school/university

*** agency — formulate/update "current plan" via passive PS, then execute; only added once all of the above is in place

*** NB the AGI's underlying hardware, the individual IDA algorithms, and the "current plan" are all formally verified[44]

[43] the human-preference-based solution to the *AI control problem* on which this final goal is based is discussed at length in [30]

[44] broadly consistent with Stuart Russell's concept of *Well Founded AI* [31]

### 2.4 Going neurosymbolic

Many aspects of the algorithmic implementation of verification (Section 2.2.5), deduction (Section 2.2.6), abduction (Section 2.2.7), induction (Section 2.2.8), and invuction (Section 2.2.10.1) may be implemented as some kind of NN[45].

Any AGI constructed in this fashion would bear a striking resemblance to the ideas propounded by Daniel Kahneman in his bestselling book *Thinking, Fast and Slow* — the low-level (connectionist) components would broadly correspond to Kahneman's *System 1*, and the high-level (symbolic) components would broadly correspond to Kahneman's *System 2*.

This combined high- and low-level approach to AGI is generally referred to as the *neurosymbolic* approach.

*** TODO

[45] *without* adversely impacting the quality of problem solutions

## 3 Conclusion

> It seems a pity, but I do not think that I can write more. For God's sake, look after our people. [32]

*** TODO

## 4 Acknowledgements

*** TODO — list and thank informal reviewers here

# BIBLIOGRAPHY

## References


[1] A. M. Turing. Computing Machinery and Intelligence. *Mind*, 49:433–460, 1950.

[2] Edward A. Feigenbaum and Pamela McCorduck. *The Fifth Generation: Artificial Intelligence and Japan's Computer Challenge to the World*. Addison Wesley, 1983.

[3] Ben Goertzel and Cassio Pennachin, editors. *Artificial General Intelligence*. Springer, 2007.

[4] I.J. Good. Speculations Concerning the First Ultraintelligent Machine. *Advances in Computers*, 6:31–88, 1966.

[5] Roman V. Yampolskiy. *Artificial Superintelligence: A Futuristic Approach*. CRC Press, 2016.

[6] Eliezer Yudkowsky. We're All Gonna Die. `https://www.youtube.com/watch?v=gA1sNLL6yg4`, 2022.

[7] Paul R Halmos. *Naive Set Theory*. D. Van Nostrand Company, 1960.

[8] A.A. Fraenkel, Y. Bar-Hillel, and A. Levy. *Foundations of Set Theory*. Elsevier, 1973.

[9] Herbert B. Enderton. *Elements of Set Theory*. Academic Press, 1977.

[10] Elliott Mendelson. *Mathematical Logic*. Wadsworth & Brooks/Cole, 3rd edition, 1987.

[11] Dominic Walliman. The Map of Mathematics. `https://www.youtube.com/watch?v=OmJ-4B-mS-Y`, 2017.

[12] Joseph R. Shoenfield. *Mathematical Logic*. Association for Symbolic Logic, 1967.

[13] Jon Barwise. An introduction to first-order logic. In Jon Barwise, editor, *Handbook of Mathematical Logic*. North-Holland, 1977.

[14] Volker Halbach. *The Logic Manual*. Oxford University Press, 2010.

[15] V.J. Rayward-Smith. *A First Course in Formal Language Theory*. Blackwell Scientific Publications, 1983.

[16] Alfred Tarski. The concept of truth in formalized languages (expanded English translation in Tarski 1983, pp. 152–278). *Wydzial III Nauk Matematyczno-Fizycznych*, 34, 1933.

[17] Alfred Tarski. *Logic, Semantics, Metamathematics: Papers from 1923 to 1938 (second edition)*. Oxford University Press, 1983.

[18] Francis Jeffry Pelletier and Allen Hazen. Natural Deduction Systems in Logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2021 edition, 2021.

[19] Michael Huth and Mark Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge University Press, 2004.

[20] Bertrand Russell. On Denoting. *Mind*, 14):479–493, 1905.

[21] Bertrand Russell. *Introduction to Mathematical Philosophy*. George Allen and Unwin, 1919.

[22] Peter Ludlow. Descriptions. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2022 edition, 2022.

[23] Doug West. *Introduction to Graph Theory*. Pearson, 2017.

[24] Ulf Grenander. *Lectures in Pattern Theory, Volume I — Pattern Synthesis*. Springer-Verlag, 1976.

[25] Ulf Grenander. *Lectures in Pattern Theory, Volume II — Pattern Analysis*. Springer-Verlag, 1978.

[26] Ulf Grenander. *Lectures in Pattern Theory, Volume III — Regular Structures*. Springer-Verlag, 1981.

[27] Ulf Grenander. *Elements of Pattern Theory*. Johns Hopkins University Press, 1996.

[28] Ulf Grenander and Michael Miller. *Pattern Theory: From Representation to Inference*. Oxford University Press, 2007.

[29] David Mumford and Agnès Desolneux. *Pattern Theory: The Stochastic Analysis of Real-World Signals*. Oxford University Press, 2007.

[30] Stewart Russell. *Human Compatible: AI and the Problem of Control*. Allen Lane, 2019.

[31] Stuart Russell. Well Founded, Human Compatible AI: Some Thoughts. `https://www.youtube.com/watch?v=mYOg8_iPpFg`, 2022.

[32] Robert Falcon Scott. Sledging diary ('Vol. III'); last entry, 29 March, 1912. `https://www.bl.uk/collection-items/captain-scotts--diary`, 1912.